**POM class 8 – Contracting**

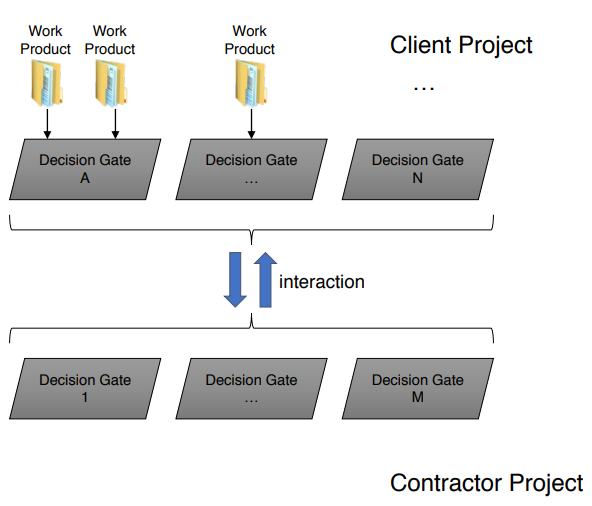
Questions to be answered:

* Expected deliverables
* What is the schedule
* Price
* Responsibilities of the contractor and the client
* What about change management

Important:

* Contract defines the basics of a project (requirements, money, time)
* Defines basic rules of collaboration
* How to deal with uncertainties (change, innovation)

There are always two projects (Client Project, Contractor Project)



**Decision Gate**:

* Milestone and Quality Gate
* Requires completed work products to determine project progress
* A project plan is a sequence of decision gates

**Client** **Project**

* Get a system

**Contractor** **Project**

* Build a system

**Contracting in the Project Life Cycle: Roles**

Procurement, acquisition and bidding -> Project Roles

**Client** (also: Customer, Purchaser, Contracting Entity)

* Define initial requirements
* Request proposals
* Monitor proposal projects
* Acceptance testing

**Contractor** (also: Supplier)

* Submit offers (bidding)
* Ship results (software -> concrete deliverable, services)

**internal** **project** **without** **any contract**…

* Department A orders a software, department B develops the software •
* Who is the client? Who is the contractor?
* How do they define the agreement?

**Influence Factors**

Projects have different influence factors -> also relevant for the contracting procedures

Main influence factors:

* Goals
* Business Case
* Initial estimations
* Legal restrictions

**What’s in a contract?**

* Services and responsibilities, incl.
  + Functional requirements
  + Deadlines (incl. procedures for delays)
  + Quality requirements (non-functional requirements)
  + Price and payment schedule
  + Acceptance
* Warranty and liability
* Provisions
* Copyrights
* Regulations and standards, compliance
* Project context variables and formal legal requirements need to be balanced in order to create an environment for the project
* A project usually starts with a dialog (acquisition)…
* Acquisition strategies:
  + Pro-active
  + Re-active
  + Client request/request for an offer
  + Call for bids/submission #
* Plan the offer’s development as a project
* A contract is always concluded based on the accepted offer

Acceptance procedure should define

**Contractor**: Shipping

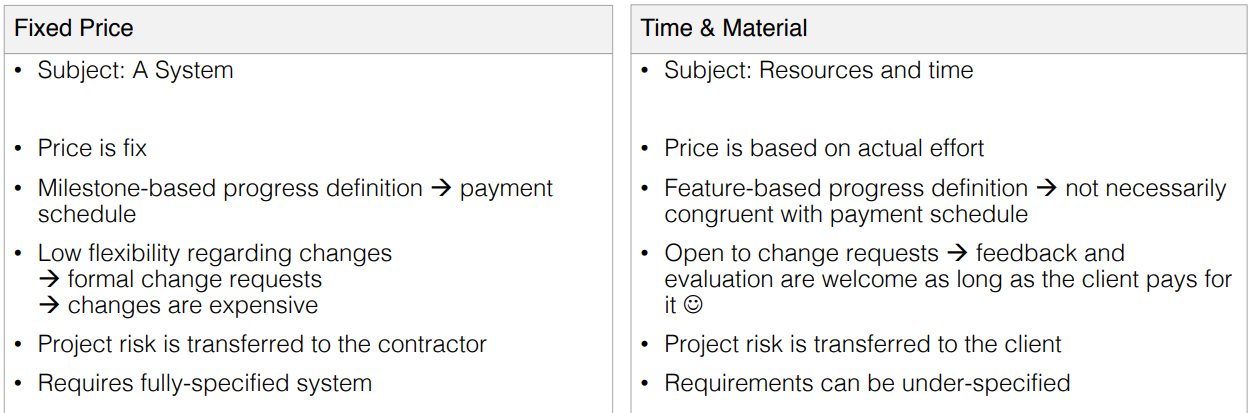
**Client**: Acceptance test, incl.

* Who is authorized to accept?
* Formal or silent acceptance? (acceptance slot, slot for correcting measures)
* Partial delivery
* Delivery dates

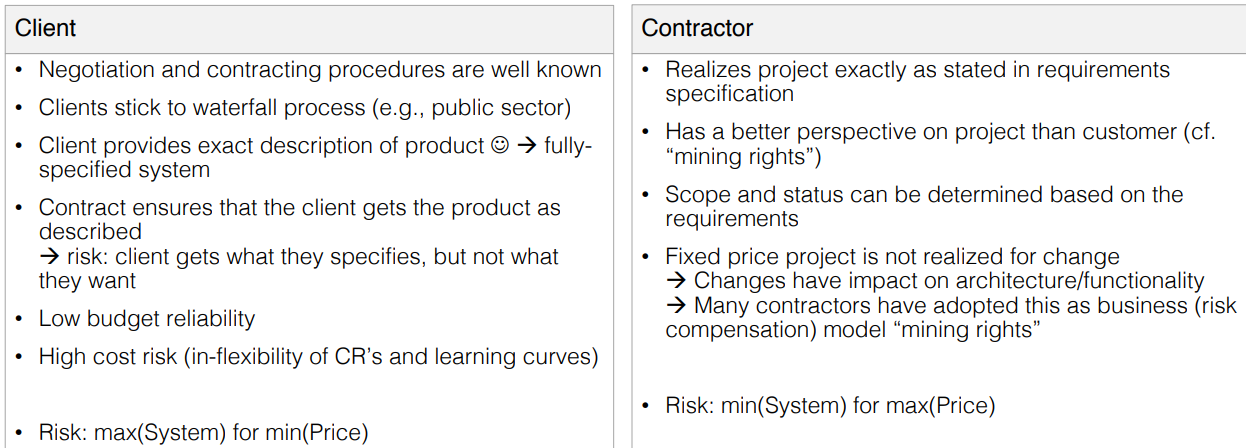
Acceptance tasks

* Check for completeness: “Was everything shipped as expected?”
* Verification of the project results: “Was the software developed right?”
* Validation of the project results: “Was the right software developed?”

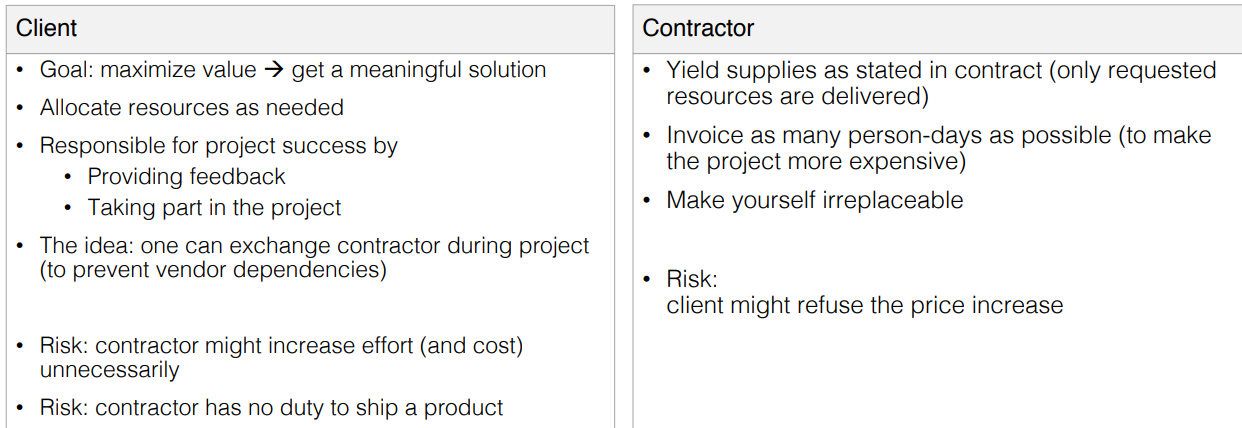
Contract Types



Fixed



Time & Material



Contracts should be designed according to the “**win-win”** interaction strategy: • Fair contract • Allows for an open atmosphere in the project

**Why is it necessary to have some alternatives**

* Companies mix processes…
  + Independent from company size and from sector
* For different reasons
  + Dependable systems and agile development practices
  + Dealing with requirements; keep flexibility
  + Connect host organization and team
* Consequence: The way contracting is implemented changes as well

**Practices**

Several practices exist to make contracts more flexible and to adopt agile methods

* Maximum price
* Change for free
* Exit point/Money for nothing

**Maximum Price**

Idea: work according to time & material approach, but set a limit

* Define and estimate the system
* Define the max. price for the system
* Define the cost per unit (e.g., story point)

**Change for Free**

**Idea**: client is participating in the project all the time

* Gives continuously feedback
* Has learning curves (better understands the system)
* Allow the client to add features, but
* Identify other features that that be removed/stalled
* Update the contract accordingly

Example: Given n features for a Sprint. Client decides to add a new feature x to the next Sprint. As compensation, a feature y is removed. Removal of y is mentioned in an updated contract

**Exit Points/Money for Nothing**

**Idea**: define an approach in which a project can be stopped, but good performance is awarded

* Define exit points (when and under which conditions to stop a project)
* Define payments to “compensate” or award early termination

**Example** (**exit** **point**): Given a project has 6 planned sprints

* Exit point: initial phase = 2 sprints for n story points
* Contractor is paid for 2 sprints -> if no exit: whole project budget
* If exit, contractor can demand compensation

**Example** (**money** **for** **nothing**): Given a project is finished early

* Contractor can be awarded with extra money, e.g., 10k for every month saved (practice often applied in construction projects)

**Combined Practices**

Idea: combine fixed price and time & material, and other practices into a defined, but flexible contract

Basic instrument: fixed price contract (maximum price)

Work is done and paid according to time & material

Exit points are defined (incl. cash flow, exit conditions)

Money for nothing is implemented to award good performance

**How to handle failure?**

Contract-relevant failure in a project can occur for many reasons

It is important to be prepared, for this:

* Define **Warranty**: when do liability and damage compensation start?
* Define **Ownership**: who owns the project results?
* Define **Copyright**: who is authorized to publish and use project results?
* Define **Escalation**: don’t go to court immediately; give mediation a chance

**Contracting: Important to know**

Contracts define the basic rules of collaboration – they set the stage…

They **define:**

* Functionality/deliverables and their respective quality
* Responsibilities
* Time/Schedule
* Money

Two **basic** types:

* Fixed price
* Time & material

Several **additional** **practices** exist to adopt flexibility and to reflect actual software development business (changing requirements, technology, etc.)

Contracts **define** **the** **rules** -> It’s not the goal to outrun the partner

Contracts **need** a **balance** (keyword: risk share)

Contracts **need** **to** **be** **fair** -> all participating parties have rights and duties